

PROMS, PALS, FIFOs, and Multipliers Team up to Implement Single-Board High-Performance Audio Spectrum Analyzer

Richard Wm. Blasco

Introduction

Digital signal processing (DSP) is rapidly becoming a popular alternative to analog techniques in high-performance applications. However, single-chip DSP devices provide only limited capability. Bit-slice processors can improve performance, but occupy excessive board space. Improved efficiency can be achieved with customized architectures, but both the single-chip and bit-slice approaches impose limitations on the designer's flexibility.

Now programmable array logic (PAL) devices, PROMs, first in-first out buffers (FIFOs), and multiplier chips can team up to implement customized architectures while keeping total package count equivalent to bit-slice solutions. The PAL devices, packaged in 20-pin skinny-DIPs, occupy less board space than 40-pin bit-slice devices. Because the PAL functions are customized by the user, PALs can provide DSP power equivalent to the bit-slices, even though the functional density is slightly less. Users get only the functions they need, and save board space as a result.

This approach is illustrated by a high-performance audio spectrum analyzer. This circuit can analyze high-fidelity audio signals with a resolution of 20 Hz and an input bandwidth of 20-kHz. It is useful in production test, performance evaluation, or adjustment of high-fidelity audio equipment. The analyzer provides a swept generator output for rapid analysis of audio filter frequency response.

The design techniques used to implement the analyzer are quite general, and can be applied to a wide variety of DSP tasks. An understanding of the approach used will suggest solutions to a number of DSP problems. The architecture chosen for the spectrum analyzer is controlled by a microprogram stored in PROM. Many other applications can be accommodated by changing the microprogram. The high performance of this architecture provides an attractive price/performance alternative to other DSP approaches.

Spectrum Analyzer Functions

The spectrum analyzer requires many of the functions commonly used in DSP. Fig. 1 shows the analyzer functions. An input signal is mixed with a swept audio sinewave oscillator. Mixing is accomplished by multiplying the input signal by the sinewave. From basic trigonometry,

$$\cos w_1 t \times \cos w_2 t = \frac{1}{2} \cos (w_1 + w_2)t + \frac{1}{2} \cos (w_1 - w_2)t \quad (1)$$

The mixing process generates two new sinewaves whose frequencies are the sum and difference of the input sinewave frequencies. When the sinewave oscillator matches the fre-

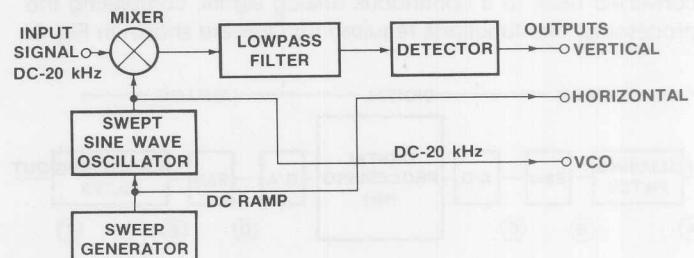


Figure 1. Spectrum Analyzer Functions

quency of an input signal component, a DC term is generated in proportion to the amplitude of that component.

The DC term is extracted by a narrow lowpass filter. Due to the finite bandwidth of this lowpass filter, output signals whose frequencies fall within the filter passband also appear at the filter output. As a result, the analyzer output will represent the energy contained in a range of frequencies, from the sinewave frequency minus the filter cutoff frequency, to the sinewave frequency plus the filter cutoff frequency. The effective bandwidth of the analyzer is twice the lowpass filter bandwidth.

A detector converts the lowpass filter output to a DC voltage representing the total energy in the filter passband. If this DC voltage is plotted on a vertical axis with the sinewave oscillator frequency (represented by the sweep voltage) controlling the horizontal axis, the spectrum of the input signal results.

Other mixing schemes can be used to extract the spectrum. However, this "direct conversion" approach has two significant advantages. As shown in Fig. 2, the swept oscillator output can be used to plot the frequency response of an audio filter. Other schemes require additional mixing to achieve the same result.

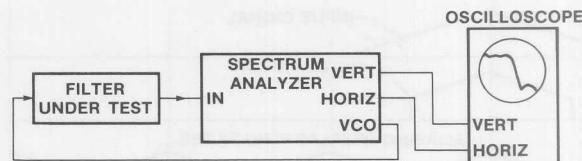


Figure 2. Filter Test Mode Setup

The direct conversion scheme confines the frequencies of all signals following the mixer to the lowpass filter bandwidth. Limiting the signal bandwidth has great benefit when the analyzer is implemented digitally. This benefit can be better understood with a brief review of DSP theory.

Digital Signal Processing Theory Review

Digital signal processing is accomplished by first converting the continuous analog input signal to a series of digital numbers. The digital numbers are then manipulated to perform the required signal processing. The processed digital numbers are converted back to a continuous analog signal, completing the processing. The functions required for DSP are shown in Fig. 3.

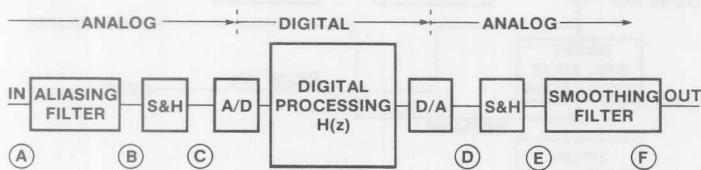


Figure 3. DSP Functions

Sampling

Representing a continuous input signal would require an infinite array of digital numbers. A finite collection of digital numbers can be obtained by considering the signal amplitude at discrete, periodic points in time. This process is called **sampling**, and is equivalent to multiplying the input signal by a periodic train of impulses of unit amplitude. The **sampling theorem** states that the input signal can be reconstructed without distortion if the input is bandlimited to contain no frequency components greater than half the sampling frequency. The sampling theorem means that the discrete samples completely represent the input signal, as long as the bandwidth constraint is met.

Aliasing

What is really happening during the sampling process? Consider the Fourier series representation of a periodic unit impulse train. It can be shown that:

$$f(t) = \sum_{k=-\infty}^{\infty} \cos(2\pi k f_s t), k = 0, 1, 2, 3, \dots \quad (2)$$

where $f_s = \frac{1}{\text{sample period}}$

The periodic impulse train is equivalent to a series of sinusoids consisting of all harmonics of the sampling frequency, including

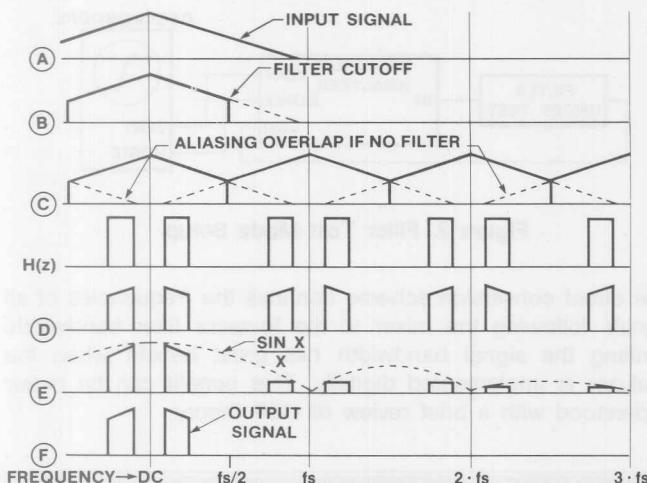


Figure 4. Aliasing Spectra of Figure 3 DSP Functions

a DC term. Recalling Eq. (1), all possible sum and difference frequencies will be generated when the impulse train and the input signal are multiplied. This process is shown graphically in Fig. 4. Observe that if the input contains frequencies greater than half the sampling frequency, the spectra in Fig. 4 will overlap. This overlap phenomenon is known as **aliasing distortion**, and introduces noise in the signal.

Another consequence of the sampling process is that high-frequency signal components near a harmonic of the sampling frequency will be mixed to produce new signal components near DC. These new components have frequencies within the desired signal passband, but are really "alias" high-frequency components. The phenomenon is called **aliasing**.

To eliminate the undesirable effects of aliasing, a continuous analog lowpass filter is placed before the sampler. This **aliasing filter** removes frequency components beyond the $f_s/2$ limit.

Quantizing

The input samples are converted to a series of digital numbers by an analog-to-digital (A/D) converter. The A/D converter operates by **quantizing** the continuous sample amplitude into a finite number of amplitude ranges, and then assigning a digital number to represent the quantized amplitude value. As might be expected, this process introduces noise in the signal, known as **quantization distortion**. The quantization distortion is in the form of a "white" or broadband random noise, whose RMS amplitude is:

$$\sigma^2 = \frac{1}{12} 2^{-2b} \quad (3)$$

where b is the number of bits in the output digital word, excluding the sign bit

The effect of aliasing on quantization noise is to alias high frequency noise components to the DC to $f_s/2$ range. The resulting noise spectral density is equivalent to a white noise of amplitude σ^2 , bandlimited to $f_s/2$.

Dynamic Range

The A/D output contains a finite number of bits. **Dynamic range** is defined as the ratio of the maximum to minimum signal amplitude that can be represented by the digital numbers. Dynamic range is determined by the number of bits in the digital numbers, and by the noise "floor."

For a digital number containing b bits plus a sign bit, the dynamic range would be:

$$\text{Dynamic range (dB)} = 10 \log_{10} 2^{-2b} \quad (4)$$

The noise floor is the sum of all noise components that can appear at the DSP output. The primary noise factors are quantization noise and limit cycle noise (to be discussed shortly). Digital filtering will affect the noise floor by eliminating components of the noise signal. For example, the quantization noise at the DSP output is:

$$N_Q (\text{dB}) = 10 \log_{10} \left[\frac{\sigma^2 \text{ BW}}{f_s/2} \right] \quad (5)$$

where BW is the net bandwidth of the digital filters

The noise components are uncorrelated, and are therefore combined by adding the power of each noise component. Remember that

$$\text{Power (absolute)} = \log_{10}^{-1} \left[\text{Power (dB)/10} \right] \quad (6)$$

The resulting dynamic range is:

$$\text{Dynamic range (dB)} = 10 \log_{10} \frac{0.5}{\Sigma \text{ Noise Power}} \quad (7)$$

where 0.5 = the maximum mean-squared amplitude

The overall dynamic range is the lesser of the result given by Eq. (4) or Eq. (7). In a practical system, the width of the digital numbers can vary. The dynamic range is usually calculated for all critical points in a digital system, with the overall dynamic range being the worst case value.

Digital Processing

The digital numbers from the A/D converter are manipulated to process the signal. Carrier generation, filtering, and nonlinear operations are performed by appropriate "number crunching".

Generation of sinusoidal carriers is easily accomplished using a linear ramp function (digital up/down counter) and converting the results to sinusoidal samples using ROM lookup tables. Alternately, recursive equations can produce the desired carriers.

Nonlinear operations on the digital numbers must be handled with care. Since aliasing is always present in the sampled domain, harmonics generated by nonlinear operations can alias to lower frequencies. The aliasing occurs "immediately," since it can be shown that performing a nonlinear operation in the sampled domain is equivalent to first performing the nonlinear operation on a continuous signal and then sampling the result without bandlimiting the sampler input.

The sampling rate can be changed to improve the efficiency of the digital processing. For example, discarding every other digital number would reduce the effective sampling rate by a factor of two. If the processing at the higher sample rate includes digital aliasing filters to remove components greater than half the lower sample rate, the requirements of the sampling theory are still met. The sampling rate can be increased by repeating digital sample values. This repetition is equivalent to a "sample and hold" operation, and modifies the signal spectrum by

$$F'(\omega) = F(\omega) \times \frac{\sin(\omega T/2)}{\omega T/2} \quad (8)$$

where $\omega = 2\pi \times \text{freq}$

T = input (longer) sample period

The effects of changing the sampling rate are best determined by plotting the resulting aliasing spectra.

Digital Filtering

Digital filtering is accomplished using multiplication, addition, and delay. For example, consider the biquadratic filter section in Fig. 5. If z^{-1} is defined to be a unit sample period delay operator, then the input-to-output transfer function of the biquadratic section is:

$$H(z) = \frac{1 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (9)$$

The biquad sections can be cascaded to implement higher-order filters.

The Laplace transform of a unit delay is e^{-sT} , where T is the delay period. Remember that z^{-1} represents an inverse operator, so that $z \times z^{-1} = 1$. Thus,

$$z = e^{-sT}, \text{ where } s = \alpha + j\omega \quad (10)$$

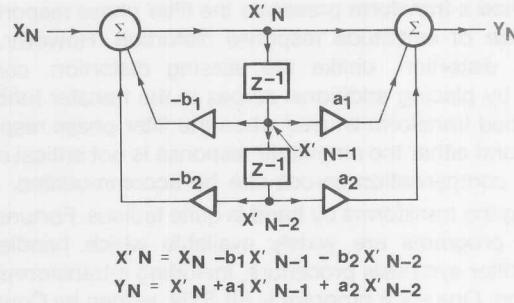


Figure 5. Digital Biquadratic Filter Section

Digital filter poles and zeroes (in the z -plane) can be mapped into the s -plane to determine the equivalent analog filter function, and vice-versa. The digital filter section of Eq. (9) corresponds to an analog biquadratic filter section with,

$$H(s) = \frac{s^2 + \alpha_0 w_0 s + w_0^2}{s^2 + \alpha_1 w_1 s + w_1^2} \quad (11)$$

However, the periodic nature of the e^{sT} function causes the digital filter passband to repeat periodically. The effect is the same as aliasing. The analog filter response is mixed with the sampling frequency harmonics to generate the true digital filter response.

Designing Digital Filters

How does one go about designing a digital filter? One approach is to perform a least mean squared error optimization using a computer. The desired function is specified, and the computer adjusts the a_n and b_n values until the desired response is achieved.

A second approach is to design an equivalent analog filter and then convert that design to a digital filter. This approach has great merit, since analog filter design theory is well developed. However, the digital passband will be distorted if the analog equivalent filter has significant response to frequencies greater than $f_s/2$. The aliased passbands overlap at that point.

To circumvent this problem, the analog filter function can be modified to compensate for the aliasing effects. The analog transfer response is modified using several transforms to compensate for aliasing. Unfortunately, the nature of the s -plane to z -plane mapping is such that no transform can compensate for all aliasing effects without introducing other forms of distortion.

The **standard (or impulse-invariant) z-transform** represents a direct mapping to the z -plane. No frequency, amplitude, or phase distortion is introduced, but aliasing effects are not compensated. This transform should be used when the analog filter has negligible response to frequencies greater than $f_s/2$.

The **bilinear z-transform** preserves the filter amplitude response in the presence of aliasing. However, the bilinear transform introduces a distortion or **warping** of the frequency axis. As a result, only the filter cutoff frequency can be accurately transformed, using a **pre-warping** technique. Frequencies within the filter passband remain warped, introducing phase distortion in the digital filter response. The bilinear transform is used when the filter amplitude response is more critical than the phase response.

PROMS, PALS, FIFOs, and Multipliers

The **matched z-transform** preserves the filter phase response at the expense of amplitude response distortion. However, this amplitude distortion, unlike the aliasing distortion, can be corrected by placing additional zeroes in the transfer function. The matched transform is used when the filter phase response is critical, and either the amplitude response is not critical or the additional compensation zeroes can be accommodated.

Performing the transforms by hand is quite tedious. Fortunately, computer programs are widely available which handle the complete filter synthesis procedure, including z-transforms and pre-warping. One such program is FILSYN, written by Compact Engineering, Inc. and available on the National CSS timeshare computer service. Once the user has determined the desired filter response and z-transform method, FILSYN will perform all necessary synthesis to generate the digital filter coefficients.

Limit cycle noise

An effect of using digital numbers with a finite number of bits is the generation of quantization noise. When implementing digital filters, the quantization noise introduces oscillations that are analogous to ringing in analog filters. These oscillations are called **limit cycles**. The limit cycles generate a noise which peaks at frequencies corresponding to the filter pole frequencies. The noise power is roughly proportional to pole Q. Limit cycle noise for a second order filter section of Eq. (11) is given by (Ref. 1):

$$N_L(\text{dB}) = 10 \log_{10} \left(\frac{2}{12} \frac{-2b}{1-r^2} \frac{1+r^2}{r^4 + 1-2r^2 \cos 2w} \right) \quad (12)$$

where b = number of digital number bits (excluding sign bit)

$$\text{pole freq.} = w_1$$

$$\text{pole } Q = 1/\alpha_1$$

$$w = 2\pi \frac{\text{pole freq.}}{f_s} \quad r = \exp \left(\frac{-w}{2 \cdot \text{pole } Q} \right)$$

The limit cycle noise must be calculated for each complex pole pair, and adjusted to reflect the response of subsequent filter stages to the limit cycle frequency. Computer programs like FILSYN can calculate limit cycle noise power, including all of these considerations.

Output signal reconstruction

Once manipulation of the digital sample numbers is complete, the resulting digital numbers must be converted back to a continuous analog signal. Referring back to Fig. 3, a digital-to-analog (D/A) converter transforms the digital numbers to a series of analog output pulses.

A sample-and-hold (S&H) circuit eliminates transients that are introduced during the D/A conversion process. The spectrum of the S&H output is modified as follows:

$$\text{S&H } F'(\text{jw}) = F(\text{jw}) \frac{t \sin(\text{wt}/2)}{T} \frac{\text{wt}/2}{\text{wt}/2} \quad (12)$$

where t = hold time T = sample period

An output smoothing filter completes the reconstruction by removing all components with frequencies greater than $f_S/2$. The smoothing filter is often optional, depending on the importance of removing the high-frequency output components.

The spectral effects of reconstruction are shown in Fig. 4.

DSP Components

Monolithic Memories, Inc manufactures a broad line of bipolar components ideally suited for DSP applications. In defining a suitable architecture for the spectrum analyzer, a familiarity with these components is essential.

Multipliers

Monolithic Memories offers three dedicated multiplier chips, with more to be announced in the near future. The chief characteristics of the Monolithic Memories' multipliers are given in Table I.

The 67508 and 67516 are sequential multipliers, while the 67558 is a full parallel multiplier. The parallel unit is considerably faster, but the sequential units provide accumulation and division as well as multiplication functions. The 67516 is particularly attractive, in that a complete 16-bit multiply/divide/accumulate function is provided in a single package.

The instruction set of the 67516 is shown in Table II. The 67516 multiply time of 1.25 μs may appear slow when compared to the multipliers in the single-chip DSP devices, but the added flexibility of the PAL approach more than compensates for the slightly reduced multiplier speed. In fact, this analyzer provides over three times the input bandwidth (determined by the input sampling rate) of comparable analyzers using the AMI S2811 or NEC $\mu\text{PD}7720$ and over six times the input bandwidth of an analyzer using the Intel 2920.

PALS

Monolithic Memories' programmable array logic (PAL) chips provide an excellent alternative to TTL when implementing random and control logic. The PAL is similar in concept to the PLA, in that logic is implemented by using an AND-OR matrix array. In the PAL, however, only the AND matrix is programmable. The OR matrix is fixed. The slight reduction in flexibility allows the addition of clocked registers, bidirectional input/output pins, exclusive-OR, and arithmetic functions on various PAL chips, all with faster propagation delay than PLAs. The PAL family is summarized in Table III.

The PALs are packaged in 20-pin "skinny dips." PC board area is minimized. The wide variety of PAL functions permits architecture optimization with a minimal total package count. PAL functional density is sufficient to compete favorably with bipolar bit-slice devices in providing a good function/speed/package count balance.

TABLE I
Monolithic Memories Multipliers

P/N	BITS	OPERATION	MPY TIME	PKG	No. PKGS. FOR 16x16	16x16 MPY TIME
67508	8x8	SEQ.	750 ns	24-DIP	1	3500 ns
67516	16x16	SEQ.	1250 ns	24-DIP	1	1250 ns
67558	8x8	PAR	100 ns	40-DIP	10*	184 ns*

*includes partial product adder matrix using Monolithic Memories' SN74S381 and SN74S182 devices

PROMS, PALS, FIFOs, and Multipliers

TABLE II
MMI 67516 INSTRUCTION SET

A0 - A2 SEQUENCE	OPERATION	CLOCK CYCLES
	0 X1Y	9
	1 -X1Y	9
	2 X1Y + Kz, Kw	9
	3 -X1Y + Kz, Kw	9
	4 Kz, Kw/X1	21
5/6 0	XY	10
5/6 1	-XY	10
5/6 2	XY + Kz, Kw	10
5/6 3	-XY + Kz, Kw	10
5/6 4	Kw/X	22
5/6 5	Kz/X	22
5/6 6 0	XY + Z	11
5/6 6 1	-XY + Z	11
5/6 6 2	XY + Kz • 2 ⁻¹⁵	11
5/6 6 3	-XY + Kz • 2 ⁻¹⁵	11
5/6 6 4	Z, W/X	23
5/6 6 5	Z/X	23
5/6 6 6 0	XY + Z, W	12
5/6 6 6 1	-XY + Z, W	12
5/6 6 6 2	XY + W _{sign}	12
5/6 6 6 3	-XY + W _{sign}	12
5/6 6 6 4	W/X	24
5/6 6 6 5	W _{sign} /X	24
	7 Read Z	1
7 7	Read Z, W	2
5 7	Round, then Read Z	2
5 7 7	Round, then Read Z, W	3

Table II Notes:

1. X, Y are input multiplier and multiplicand.
2. X1 is previous multiplier value.
3. 5/6 specifies input format. Use 5 for fractional arithmetic and 6 for integers.
4. Z, W is a double-precision number. Z is MSB. Represents addend on input and product (or accumulator sum) on output.
5. Kz, Kw represents previous accumulator contents. Kz is MSB.
6. W_{sign} is a single length signed number.
7. Each clock cycle = 125 ns for an 8-MHz clock.

FIFOs

The 67401 FIFO is a 64x4-bit device providing very fast (10 MHz) shift rates. The elastic storage provided by FIFOs permits sample rate reduction in the analyzer with minimal software overhead. Similar sample rate reductions cannot be accomplished using single-chip DSP devices, because of the very limited elastic storage provided.

The FIFOs maintain a uniform, jitter-free input and output sample rate for the analyzer. This is required by sampling theory. However, crunching of the digital numbers within the analyzer can be non-uniform, as long as sufficient FIFO elastic storage is provided. With FIFO buffers at the input and output, internal sample rates may be optimized, limited only by aliasing considerations.

PROMs

The 63xx series of bipolar programmable read-only memory (PROM) chips provide storage for filter coefficients and micro-program control. The 6309-1 PROM is a versatile 256x8-bit device, with a 70 ns worst-case access time, making it ideally suited for use in the analyzer.

Other Components

Monolithic Memories also second-sources a variety of Schottky TTL octal registers, octal bus buffers, and arithmetic chips. The spectrum analyzer can be efficiently implemented using the Monolithic Memories' components and only a handful of additional devices.

Implementing the Spectrum Analyzer

The architecture used for the spectrum analyzer is shown in Fig. 6. Input signals are digitized and buffered with FIFOs before interface with a common 16-bit data bus. The 16-bit arithmetic unit (AU) provides multiply and accumulate operations. A 16-bit wide RAM stores intermediate results. A 16-bit temporary register facilitates z^{-1} delays and data movement. Outputs are provided using a D/A converter and S&H circuits.

TABLE III
Monolithic Memories PAL-20 Family

P/N	INPUTS	OUTPUTS	BIDIRECTIONAL I/O	REGISTER BITS	FUNCTIONS
PAL10H8	10	8	—	—	AND-OR
PAL12H6	12	6	—	—	AND-OR
PAL14H4	14	4	—	—	AND-OR
PAL16H2	16	2	—	—	AND-OR
PAL16C1	16	1	—	—	AND-OR/NOR
PAL10L8	10	8	—	—	AND-NOR
PAL12L6	12	6	—	—	AND-NOR
PAL14L4	14	4	—	—	AND-NOR
PAL16L2	16	2	—	—	AND-NOR
PAL16L8	8	—	8*	—	AND-NOR
PAL16R8	8	8*	—	8	AND-OR
PAL16R6	8	6*	2*	6	AND-OR
PAL16R4	8	4*	4*	4	AND-OR
PAL16X4	8	4*	4*	4	AND-OR-XOR
PAL16A4	8	4*	4*	4	AND-CARRY-OR-XOR

*TRI-STATE OUTPUTS

PROMS, PALS, FIFOs, and Multipliers

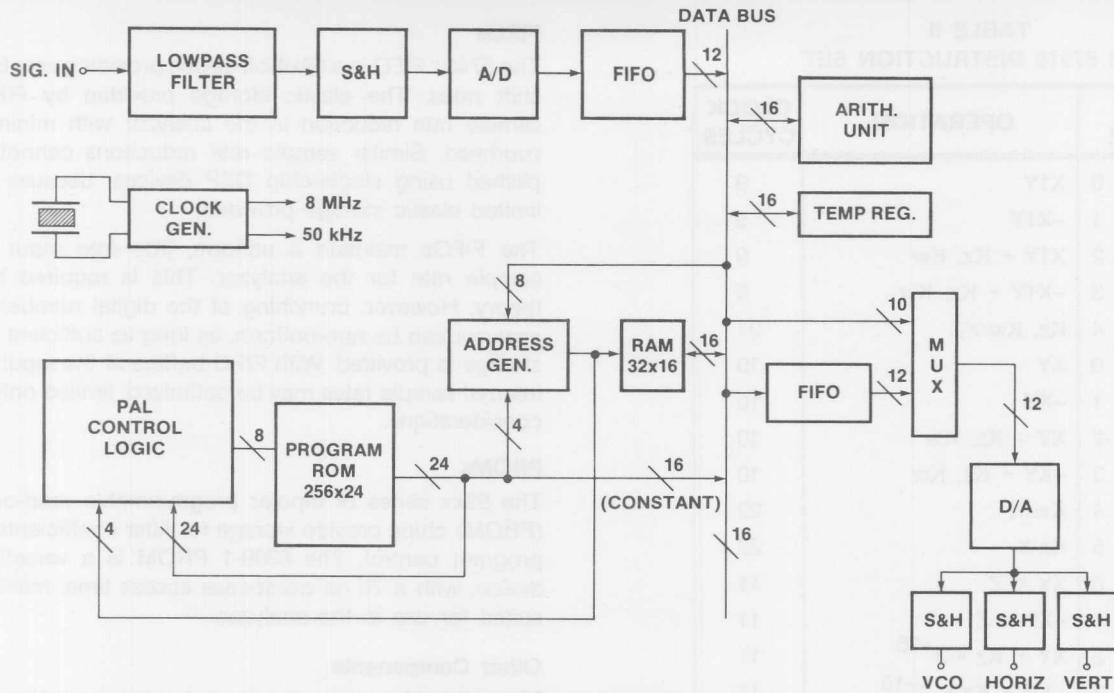


Figure 6. Analyzer Architecture

The VCO output is buffered using FIFOs to provide a uniform high-speed sample rate. The VCO output is 12-bits wide, providing a signal-to-quantization noise ratio of 91 dB, using Eqs. (5) and (7). The calculation assumes a 500-Hz bandwidth. A smoothing filter at the VCO output is not necessary. The filter test configuration of Fig. 2 allows the input aliasing filter to remove the effects of VCO high-frequency components, as long as the filter under test is a linear analog circuit.

The vertical and horizontal outputs are intended to interface an oscilloscope or X-Y plotter. The sampling of these outputs can be non-uniform, as long as the outputs track each other. The elastic storage at the input and VCO interfaces permits arbitrary non-uniform processing of the analyzer functions.

The 16-bit resolution of the internal data word provides 90 dB dynamic range according to Eq. (4), or 115 dB dynamic range according to Eqs. (5) and (7), assuming 500 Hz bandwidth and no limit cycle noise and aliasing.

Microprogram control was selected for the analyzer. PALs can efficiently implement sequential state machines. It is possible to encode all control information in the PALs, but only three packages would be saved (one PROM and two buffers). Distributing the control among several PALs would reduce flexibility and make corrections very difficult. The few extra packages required for microprogram control provide an extremely flexible architecture and greatly simplify the PAL functions.

With the theoretical background and architecture in mind, the spectrum analyzer functions can be defined in detail. The objective is to realize a circuit capable of high-resolution analysis of audio signals in the DC to 20-KHz range. Selectable bandwidth and linear/logarithmic output display are highly desirable. The detailed functions are shown in Fig. 7.

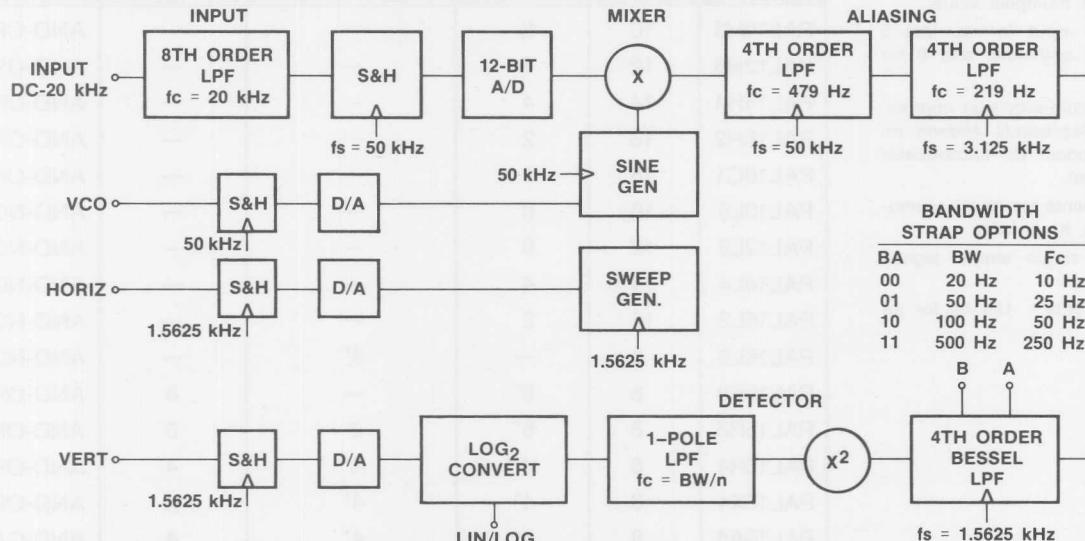


Figure 7. Detailed Functional Diagram

Input aliasing filter

An analog lowpass filter removes high-frequency components from the input signal. With a sample of 50 kHz and a maximum input frequency of 20 kHz, the lowest aliasing frequency is $(50-20) = 30$ kHz.

An eight-order Chebychev filter with 0.1 dB passband ripple will provide 44 dB of attenuation at 30 kHz, and 86 dB attenuation at 50 kHz. It is desirable to provide at least 60 dB overall dynamic range for high-performance analysis. To eliminate spurious responses above the -60 dB "floor," the input signal should have all components above 30 kHz suppressed by at least $(60-44) = 16$ dB. Most input signals will meet this requirement. If not, additional filtering must be provided.

S&H and A/D converter

The input S&H maintains a constant sampled signal level while the A/D conversion is in progress. No sin X/X correction is made for this S&H since the net effect of the S&H plus A/D action is an impulse sample at the start of the "hold" period.

The A/D conversion time should be less than 16 μ s. The A/D converter output digital number should "saturate" when the input signal exceeds the maximum level. The digital numbers should be in inverted two's complement form. The S&H acquisition time should be less than 4 μ s.

For a full 60 dB overall dynamic range, a 12-bit A/D is required. The 1980 IC Master lists several A/D converters meeting all of these requirements.

Mixer

The mixer multiplies the A/D output by the swept sinewave oscillator value. The multiplication produces sum and difference frequencies, according to Eq (1).

Two's complement fractional arithmetic is used throughout the analyzer. Multiplication cannot overflow, since all numbers are less than 1 in magnitude.

Swept sinusoidal oscillator (VCO)

A precision swept sinusoid from DC to 20 kHz must be generated to mix with the input signal. A technique particularly well suited to this application is solving the two equations:

$$\sin(x+y) = \sin x \cos y + \cos x \sin y \quad (13)$$

$$\cos(x+y) = \cos x \cos y - \sin x \sin y \quad (14)$$

These two trigonometric identities generate a new sin and cos value with y representing the phase shift per sample period. The technique is a "CORDIC" algorithm, based on coordinate rotation. Exact results are produced, but truncation and round-off errors due to the finite digital word length can cause a slow change or "drift" of carrier amplitude. Fortunately, the swept sinusoid is periodically reset in the spectrum analyzer, arresting this amplitude drift.

The VCO frequency is swept by varying the value of y. However, since Eqs. (13) and (14) require sin y and cos y, an identical CORDIC algorithm is used to obtain these values. To sweep the VCO, then, sin Δ and cos Δ are placed in RAM, selected by the bandwidth setting. These are two fixed numbers originally stored in PROM, and represent the frequency shift per sample time. Eqs. (13) and (14) are then applied to calculate sin y and cos y, which represent the desired phase shift per sample time. Eqs. (13) and (14) are executed again to generate the actual sinusoidal output.

The calculation of sin y and cos y can take place at a reduced sample rate to save processing time. Only the last execution of Eqs. (13) and (14) must be performed at the full 50-kHz sample rate.

A linear ramp is generated to provide horizontal drive for an oscilloscope or X-Y plotter. The ramp is incremented each time the sin y and cos y values are updated, tracking the VCO sweep. When the ramp value overflows, the analyzer sweep cycle is re-initialized.

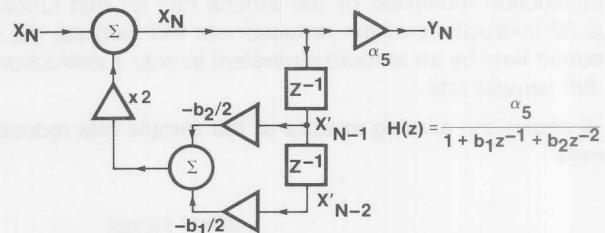


Figure 8. All-Pole Digital Filter Section

Digital filters

Fig. 8 shows the implementation of the all-pole digital filter sections. Because of the low pole Q values in all filters, the second order sections can be simply cascaded to implement high-order filters. Fig. 8 shows a technique for handling coefficients greater than 1 with fractional number representation.

Scaling must be performed to ensure maximum dynamic range. Filter sections with high-Q poles will show peaking of signals near the pole frequencies. The input to such sections must be scaled down to prevent overflow of the arithmetic. For a second-order all-pole section, this peaking factor is exactly the Q of the poles. Thus, when a given second-order section has a pole Q of 2, the input signal to that section must be multiplied by 0.5 to prevent overflow. When the Q is less than or equal to 1, no scaling is performed.

Saturation arithmetic is not provided in this architecture. Careful scaling eliminates the need for saturation arithmetic, since the A/D will saturate at a precisely known value.

The digital filters are synthesized and analyzed using the FILSYN program. FILSYN provides a limit cycle analysis of individual filter sections and of the overall filter response. FILSYN is a highly interactive program, and is quite straightforward in its use. Use of such programs permits the design of digital filters with only a brief familiarity of the theoretical concepts involved.

Aliasing filters

Two 4th-order Chebyshev filters permit reduction of the sample rate following the mixer. Each filter provides 0.3 dB passband ripple and at least 68 dB attenuation of aliasing components. The slightly high passband ripple is acceptable, since subsequent filters will dominate the composite passband shape.

The first filter permits a sample rate reduction factor of 16. It is designed with a passband cutoff frequency of 479 Hz and a sample rate of 50 kHz. Eq. (12) predicts the limit cycle noise for this filter to be -58 dB.

PROMS, PALS, FIFOs, and Multipliers

The second filter permits a second sample rate reduction factor of 2. Its cutoff frequency is 219 Hz, with a sample rate of 3.125 kHz. Eq. (12) predicts the limit cycle noise for this filter to be -83 dB. This filter also provides an additional 14 dB attenuation of the limit cycle noise generated in the first aliasing filter, reducing the limit cycle noise from the first filter to -72 dB.

These two filters permit an overall f_s reduction factor of 32 before processing the Bessel filter, detector, and linear-to-log₂ functions. This results in a very substantial throughput improvement. Net execution time is determined by the time to execute a given function multiplied by the sample rate for that function. Thus 32 instructions at the reduced rate will increase the net execution time by an amount equivalent to only 1 instruction at the full sample rate.

Fig. 9 shows the aliasing spectra of the sample rate reduction process.

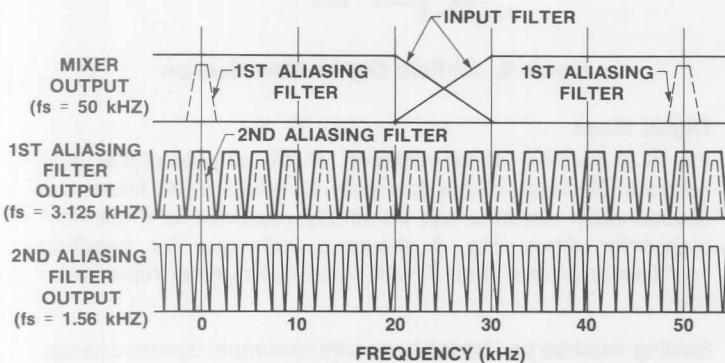


Figure 9. Aliasing Spectra for f_s Reduction

Lowpass filter

A 4th-order Bessel lowpass filter determines the overall bandwidth of the analyzer. The overall bandwidth is twice the bandwidth of this filter. Overall bandwidths of 20, 50, 100 or 500 Hz are provided by loading the proper set of filter coefficients into RAM when the bandwidth is selected.

The Bessel filter provides an optimal transient response for the analyzer. Good transient response is important, especially at narrow bandwidths, since the spectral peaks are swept with respect to the filter passband. The net effect is similar to pulsing the filter input. Because the phase response is critical, the matched-z transform is used to convert the analog Bessel design to the z-domain.

The second aliasing filter provides 3 dB of attenuation at 250 Hz. When cascaded with the Bessel filter, which also provides 3 dB attenuation at 250 Hz in the 500 Hz bandwidth mode, the response at this bandwidth is modified. However, since the overall bandwidth is relatively broad, good transient response is still achieved. Cascading these two filters provides a "transitional" filter with a Bessel response at low attenuation and a Chebyshev response at high attenuation. At bandwidths less than 500 Hz, the combination produces an optimal tradeoff between transient response and resolution.

Analysis of Eq. (12) reveals that limit cycle noise increases exponentially as the pole frequency is reduced. Operating the lowpass filter at the lowest possible sampling frequency (1.5625 kHz) minimizes limit cycle noise, in addition to improving throughput. Limit cycle noise for the lowpass filter will be -95

dB at the 500 Hz bandwidth, and increases to -67 dB at the 20 Hz bandwidth.

Detector

A square-law detector provides a DC signal corresponding to the energy at the lowpass filter output. From trigonometry:

$$(A \cos wt)^2 = \frac{A^2}{2} (1 + \cos 2wt) \quad (15)$$

The detector output contains the desired DC term and a single undesired term at frequency $2w$. If the square law is ideal (easy in the digital domain), no additional terms are produced. The elimination of harmonics ensures the accuracy of the detector with $f_s/32 = 1.5625$ kHz. The highest component is always less than $f_s/64$ with a 250 Hz maximum lowpass filter cutoff frequency. However, $2w$ can be anywhere from DC to 500 Hz as the VCO sweeps past the spectral component.

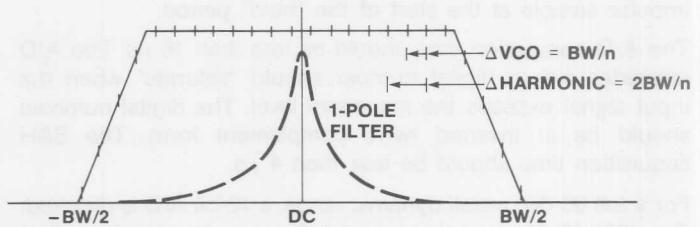


Figure 10. Detector Sweep Filtering

Fig. 10 illustrates a technique to render the effects of the $2w$ terms negligible. The analyzer passband is divided into n equal intervals. The VCO sweep rate is controlled so that the VCO sweeps BW/n per $f_s/32$ interval. The detector is followed by a single-pole lowpass filter with a 3 dB frequency of BW/n . As the VCO sweeps a component through the passband, the DC term is present in all n intervals, but the $2w$ term can affect only one interval. The worst-case DC error is $1/n$ for an ideal cutoff, and is multiplied by $(2^0 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-5.5} + 2^{-6} + 2^{-6.5} + 2^{-7} + 2^{-7.25} + \dots) = 1.85$ due to the finite 6 dB/octave rolloff of the single-pole filter. Further analysis reveals that:

$$n = \frac{1.85}{10^{e/10} - 1} \quad (16)$$

where e represents the resulting error in dB. For $e = 0.1$ dB, $n = 80$.

In the filter test mode, the signal frequency and VCO frequency are the same, forcing $w = 0$. The detector has no error in this mode, but has a 3 dB gain due to the second DC term.

The detector output represents signal energy. Each bit in the detector output word thus represents only 3 dB, and 21 bits are required to reflect a 60 dB dynamic range. Double precision arithmetic is required for the detector output and the single-pole filter. The 67516 multiplier will handle double precision calculations with a time penalty. Fortunately, the calculations to be performed are simple and the operations take place at the minimum sample rate, reducing the impact on throughput.

Linear to log conversion

The architecture of Fig. 6 is customized to provide an efficient algorithm for linear to logarithmic output conversion. The RAM address generator monitors the 8 MSBs of the data bus, and can provide a number indicating the MSB position of a positive

PROMS, PALS, FIFOs, and Multipliers

number. This output is used to retrieve a lookup table value. This value is used to scale the data word to quickly left-justify the MSB. A second 4-point lookup table is then used to improve the accuracy of the resulting \log_2 conversion. A \log_2 conversion is adequate, since:

$$\log_{10} x = \frac{\log_2 x}{\log_2 10} \quad (17)$$

Eq. (17) demonstrates that the output can be displayed in decibels by setting the oscilloscope or X-Y plotter Y-axis gain to the proper value.

Two lookup tables provide .027 dB accuracy for output values from 0 to -45 dB, and 3 dB accuracy from -45 to -84 dB. The logarithmic accuracy is limited by the 10-bit output word length to the D/A. This output can represent 0 to -84 dB in .082 dB increments. The accuracy of the 4-point lookup is therefore sufficient.

The logarithmic conversion procedure is as follows:

- If the MSB of the data word is not among the 8 MSBs into the address generator, multiply the word by $2^7 = 128$, and increment the output number by 7. Repeat until the data word is greater than 2^7 , but no more than three times. Set a flag if this step is executed more than once.

- Look up the appropriate scale factor, from 2^0 to 2^6 . Add \log_2 of the scale factor to the output word. The conversion is now accurate to 3 dB.
- If the flag was not set during step 1, multiply the data word by the scale factor to left-justify the MSB position.
- If the flag was not set during step 1, retrieve an intercept and slope value from the 8-word lookup table (four pairs available). Perform a linear interpolation using:

$$x' = a x + b \quad (18)$$

where a is the slope value
 b is the intercept value

The conversion is now accurate to .027 dB.

- Scale the result to provide 84-dB output range with a 10-bit word.
- Subtract 2^{-1} from the output to convert it to two's complement form for the D/A.

The calculations are double precision for steps 1, 2, and 3, and single precision thereafter. The conversion sequence can be bypassed using a strap option to provide a linear amplitude output from 0 to -30 dB.

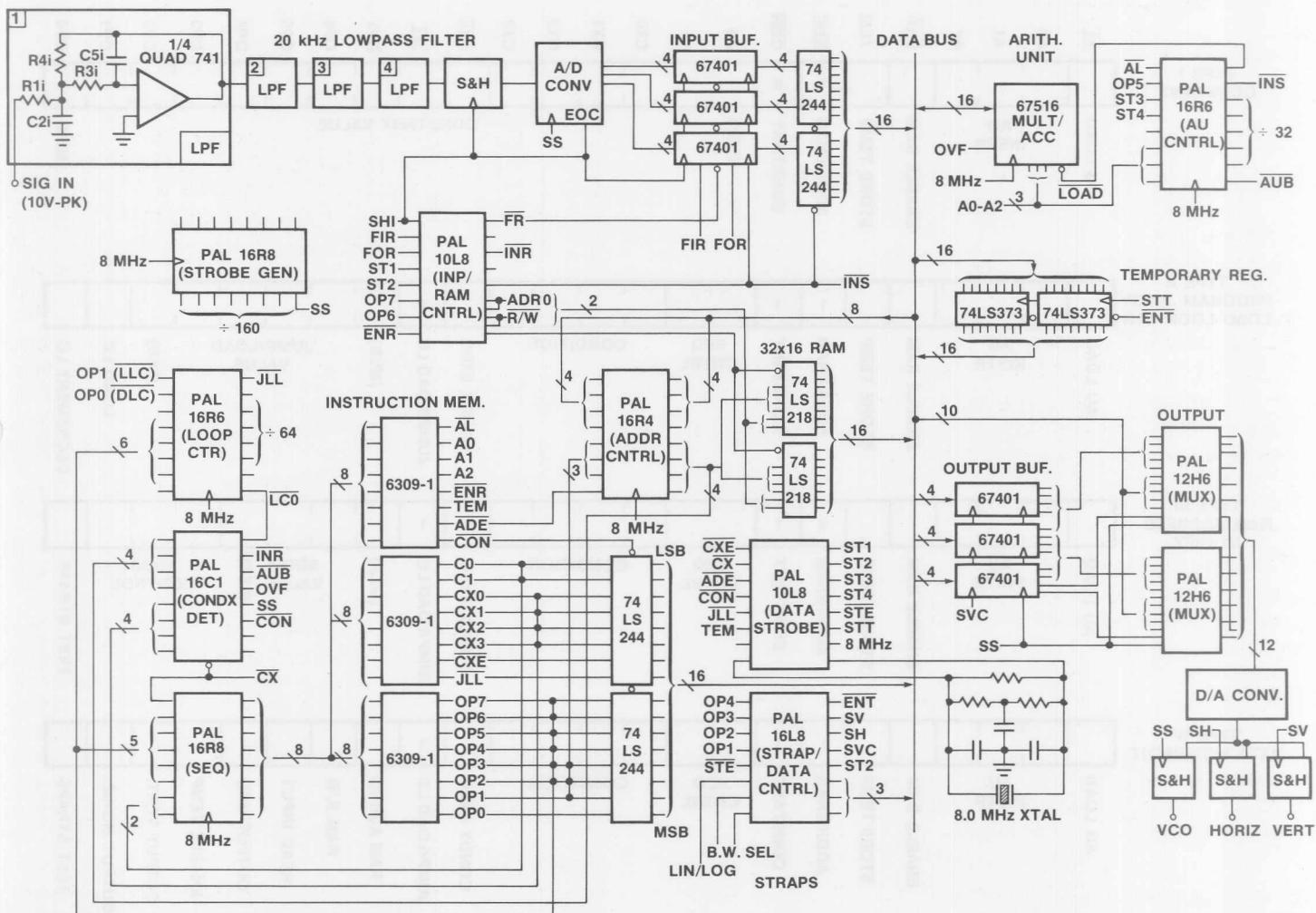


Figure 11. Simplified Schematic Hi-Fi Audio Spectrum Analyzer

PROMS, PALS, FIFOs, and Multipliers

Control logic

Fig. 11 shows a simplified schematic of the analyzer. All critical components are shown. Bypass capacitors and some component input connections are omitted for clarity.

The microprogram is stored in three 6309-1 PROMs. The microcode word formats are shown in Fig. 12. A wide, highly parallel instruction word ensures maximum flexibility and program efficiency.

Eight PAL devices interpret the instruction word and control the analyzer. Two additional PALs generate a 50-kHz strobe from the 8-MHz master clock, and implement the output D/A multiplexer. The control PALs function as follows:

Sequencer. A PAL16R8 implements an 8-bit instruction sequencer. A binary sequence would exhaust the available PAL OR-terms, so the sequencer is implemented using a 5-bit polynomial counter cascaded with a 3-bit binary counter. The sequencer performs the following operations:

C1 C0 CX Operation

0	0	X	Increment by 1 (execute next instruction)
0	1	0	Increment by 2 (skip next instruction)
1	0	0	Jump to address
1	1	0	No increment (repeat current instruction)

The \overline{CX} input conditions the sequencer. Conditional branches or skip operations can be implemented. The sequencer will increment if the conditional requirement is not met.

A 5-bit jump destination can preset the polynomial counter. The binary counter stages are set to zero during a jump operation. Every eighth sequence address is, therefore, a possible jump destination. This compromise maintains adequate program flexibility, while permitting a single PAL to implement the sequencer.

Condition detector. A PAL16C1 monitors up to twelve status flags, and generates \overline{CX} . The microcode word

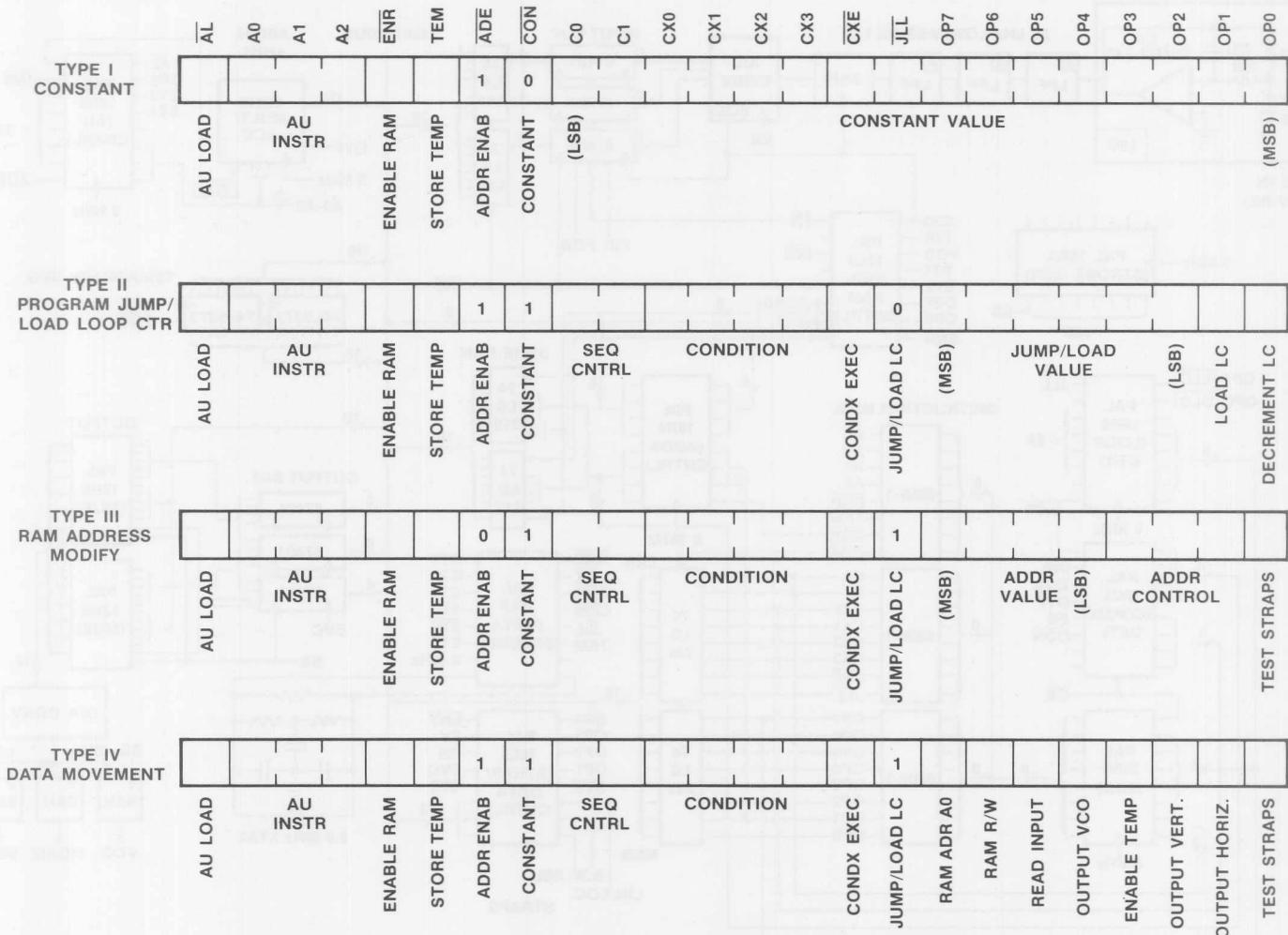


Figure 12. Microinstruction Word Formats

PROMS, PALS, FIFOs, and Multipliers

includes a 4-bit condition word, CX0 through CX3. \overline{CX} will be zero under the following conditions:

CX3 CX2 CX1 CX0 Condition for $\overline{CX} = 0$

0	0	0	0	Always (unconditional operation)
0	0	0	1	Sample strobe (SS) = 1
0	0	1	0	AU overflow (OVF) = 1
0	0	1	1	AU busy (AUB) = 1
0	1	0	0	Input sample ready (INR) = 0
0	1	0	1	Loop counter timeout (LLC) = 0
0	1	1	0	LLC = 1
0	1	1	1	Address control (AC3-AC0) = 0
1	0	0	0	AC0 = 0
1	0	0	1	AC0 = 1
1	0	1	0	AC1 = 0
1	0	1	1	AC1 = 1
1	1	0	0	AC2 = 0
1	1	0	1	AC2 = 1
1	1	1	0	AC3 = 0
1	1	1	1	AC3 = 1

The assignment is made using the flexible PAL coding, and is optimized for the analyzer. The user can select a different set of conditions by re-programming the PAL.

When the microcode represents a constant (Type I microinstruction — see Fig. 12) the CON input forces $\overline{CX} = 1$ to suppress conditional operations. \overline{CX} is also used to suppress certain strobes in the analyzer, providing conditional arithmetic operations.

Loop counter. A PAL16R6 implements a 6-bit programmable down counter. This counter controls iteration loops and provides a timeout signal to the condition detector. The counter is preset via a Type II microinstruction, and can be decremented by other Type II microinstructions. The counter will halt when zero count is reached. Up to 64 iterations can be accommodated with minimal overhead.

Address control. A PAL16R4 provides indexed addressing for the 32x16 RAM, and analyzes the 8 MSBs of the data bus for conditional operations. If D15 represents the data bus sign bit, then OP1-OP3 will provide the following functions:

OP3 OP2 OP1 AC3-AC0 Output Function

0	0	0	Clear (0000)
0	0	1	Increment
0	1	0	Decrement
0	1	1	Preset to D15-D12 (Sign + 3 MSB)
1	0	0	Preset to D14-D11 (4 MSB)
1	0	1	Preset to D11-D8 (Address load)
1	1	0	MSB position
1	1	1	No change

The \overline{ADE} input enables a change in the address word. The address word will not change if $\overline{ADE} = 1$.

The MSB position function indicates the position of the MSB for positive numbers. AC3 represents sign bit D15. This output should be zero. AC2-AC1 represent the position of the first 1 following the sign bit. Code 0000 indicates that D15-D8 are all 0.

Input/RAM control. Miscellaneous FIFO input and RAM control is provided by a PAL10L8. The 67401 FIFO includes input ready (FIR) and output ready (FOR) signals, which are latched using the input/output shift clocks to generate two flags. The first flag (FR) resets the FIFO when input ready (latched) goes low, indicating the FIFO capacity is exhausted. The latched output ready signal flag represents input sample ready (INR). The INR

flag is used as a sequencer condition to synchronize wait loops. Use of the FR and INR flags maintains proper fill of the FIFO.

The RAM address LSB (A0) and read-write line (R/\overline{W}) are decoded and latched. These signals are provided directly by Type IV microinstructions.

Notice that a clocked register function requires two PAL combinatorial outputs per bit, while a transparent latch function requires only one PAL output per bit.

Arithmetic unit control. The variety of functions listed in Table III indicate the utility of the 67516 arithmetic unit (AU). A PAL16R6 complements the 67516 to provide simplified control of the AU.

The PAL gates and AU load signal provide conditional arithmetic operations. Gating the 67516 load input will suppress the start of a new arithmetic operation. When \overline{CXE} is high, the operation is performed unconditionally. When \overline{CXE} is low, the operation is performed only if \overline{CX} is low. Combining conditional jumps and conditional AU operations provides a high degree of program flexibility.

The PAL monitors the AU instructions and generates a busy signal (AUB). A counter in the PAL keeps track of variable-length operations to provide the correct output for any instruction sequence. The AUB signal conditions the sequencer to synchronize the microprogram to the AU operation. Microprogramming is simplified as a result.

The PAL also gates the input FIFO shift out clock (\overline{INS}) to eliminate transients while providing a full 125 ns pulse for proper FIFO operation.

Data strobe generator. A PAL10L8 provides a number of transient-free, gated strobes. These strobes provide control of the analyzer data flow. The PAL gating interprets the microinstruction to determine the proper microinstruction type, and generates the strobes accordingly.

The PAL also generates an 8-mHz buffered clock, as shown in Fig. 11. The crystal oscillator circuit provides independent AC and DC feedback, permitting reliable operation with the PAL.

Strap/output sample control. A PAL16L8 generates additional control strobes for the output sample-and-hold circuits.

The PAL also provides a tri-state buffer function, connecting control straps to the data bus for certain conditional jump operations. Two straps select the desired analyzer bandwidth/sweep rate, and the third strap selects linear or logarithmic output.

Programming the PALs. The PALs provide a wide variety of control functions that are optimized by user programming. Several methods are available to allow easy PAL programming.

Programming is accomplished by blowing fuses in the PAL AND-array. Each AND-term contains the true and complement of all inputs. When no fuses are blown, the AND-term is disabled, since $A \cdot \overline{A} = 0$. AND-terms are activated by blowing fuses to disconnect unwanted inputs. The AND-terms are combined using OR, NOR, or exclusive-OR gates (depending on the PAL type) to generate the PAL outputs. The result is a sum-of-products Boolean expression representing the desired logic function.

The sum-of-products Boolean expressions are defined using standard digital logic design procedures. Combinatorial functions are plotted on Karnaugh maps and minimized. State tables and flowcharts are prepared for sequential state machines.

PROMS, PALS, FIFOs, and Multipliers

Once the state tables are minimized, the state variable transition functions are plotted on Karnaugh maps and minimized. The details of this procedure are described in the literature (Refs. 3, 4, 5).

Once the Boolean expressions have been derived, PALs may be obtained using several procedures:

- 1) Provide Monolithic Memories with a coded sheet indicating the desired fuse pattern. Blank coding sheets are available from Monolithic Memories.
- 2) Prepare an ASCII paper tape using either a "BPNF" or a hexadecimal format to represent the pattern, and mail or TWX the pattern to Monolithic Memories.
- 3) Obtain unprogrammed PALs and blow the fuses using a bipolar PROM programmer with an appropriate personality module. One such unit, designed specifically for PAL programming, is the Model SD-20 from Structured Design, Inc., 330 N. Mathilda Avenue, MS112, Sunnyvale, CA 94086.

Monolithic Memories offers PALASM, an assembler to generate PAL fuse patterns or paper tape files from Boolean expressions. This program is available from Monolithic Memories via the National CSS timeshare computer network. PALASM is an interactive, easily used tool for rapid pattern generation. The SD-20 programmer software incorporates the essential features of PALASM in its operation.

The programming procedure is thoroughly described in the **PAL Programmable Array Logic Handbook**, available from Monolithic Memories. A FORTRAN source listing of PALASM is included in the handbook.

Signal output

The VCO output must be sampled at precise intervals to avoid phase modulation effects. Three 67401 FIFOs buffer the VCO samples, which are generated during the 50-kHz input processing. A 12-bit D/A converter provides better than 91 dB signal-to-distortion ratio. The S&H circuit provides VCO outputs at precise 50-kHz intervals, and removes spikes that are generated in the D/A converter. All necessary control signals are generated by the strap/output data control PAL.

The horizontal and vertical outputs normally drive an X-Y plotter or oscilloscope. There is no need to buffer these signals as long as the two outputs track each other. The D/A used for the VCO output is shared by adding two PAL12H6 chips programmed as multiplexers. Use of PALs requires fewer packages than a TTL multiplexer. Additional S&H circuits decode the multiplexed D/A output to separate the output signals.

Microprogram

The architecture can implement a variety of DSP functions. A microprogram, stored in 6309-1 PROMs, customizes the architecture to perform the spectrum analyzer tasks. The microinstruction formats were summarized in Fig. 12. The algorithms to be implemented were discussed in the previous section. The step-by-step implementation of these algorithms is converted to a sequence of microinstructions to form the microprogram. The procedure is analogous to programming a microprocessor.

Operation of the microprogram is better understood by considering the allocation of the 74LS218 RAM locations, shown in Fig. 13. The microprogram consists of two parts. High-speed

(A4-A1)																(A0)																																		
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E,	F																																			
—	—	SIN X	SIN Y	SIN Δ	b1	b1	—	X _{N-1}	X _{N-2}	X _{N-1}	X _{N-2}	X _{N-1}	X _{N-2}	X _{N-1}	X _{N-2}	0																																		
—	—	COS X	COS Y	COS Δ	b2	b2	—	X _{N-1}	X _{N-2}	X _{N-1}	X _{N-2}	X	X	X	VCO RAMP	1																																		
CARRIER GENERATOR						LOWPASS FILTER						ALIASING FILTERS					LOWPASS FILTER																																	
(a) INPUT																																																		
<table border="1"> <tr> <td>TEMP LSB</td> <td>FLAG</td> <td>X</td> <td>X</td> <td>SCALE A</td> <td>SCALE B</td> <td>SCALE C</td> <td>SCALE D</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>0</td> </tr> <tr> <td>TEMP MSB</td> <td>OUTPUT</td> <td>X</td> <td>X</td> <td>LOG A</td> <td>LOG B</td> <td>LOG C</td> <td>LOG D</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X_{N-1} LSB</td> <td>X_{N-1} MSB</td> <td>b1</td> <td>X</td> <td>1</td> </tr> </table>																TEMP LSB	FLAG	X	X	SCALE A	SCALE B	SCALE C	SCALE D	X	X	X	X	X	X	X	X	0	TEMP MSB	OUTPUT	X	X	LOG A	LOG B	LOG C	LOG D	X	X	X	X	X _{N-1} LSB	X _{N-1} MSB	b1	X	1	
TEMP LSB	FLAG	X	X	SCALE A	SCALE B	SCALE C	SCALE D	X	X	X	X	X	X	X	X	0																																		
TEMP MSB	OUTPUT	X	X	LOG A	LOG B	LOG C	LOG D	X	X	X	X	X _{N-1} LSB	X _{N-1} MSB	b1	X	1																																		
UTILITY						LIN-LOG SCALING						DETECTOR																																						
(b) OUTPUT-1																																																		
<table border="1"> <tr> <td>TEMP LSB</td> <td>FLAG</td> <td>X</td> <td>X</td> <td>a₁</td> <td>a₂</td> <td>a₃</td> <td>a₄</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>0</td> </tr> <tr> <td>TEMP MSB</td> <td>OUTPUT</td> <td>X</td> <td>X</td> <td>b₁</td> <td>b₂</td> <td>b₃</td> <td>b₄</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>X</td> <td>1</td> </tr> </table>																TEMP LSB	FLAG	X	X	a ₁	a ₂	a ₃	a ₄	X	X	X	X	X	X	X	X	0	TEMP MSB	OUTPUT	X	X	b ₁	b ₂	b ₃	b ₄	X	X	X	X	X	X	X	X	1	
TEMP LSB	FLAG	X	X	a ₁	a ₂	a ₃	a ₄	X	X	X	X	X	X	X	X	0																																		
TEMP MSB	OUTPUT	X	X	b ₁	b ₂	b ₃	b ₄	X	X	X	X	X	X	X	X	1																																		
UTILITY						LIN-LOG INTERPOLATION						KEY: <input checked="" type="checkbox"/> SCRATCH — NOT USED <input type="checkbox"/> TABLE					X NOT AVAILABLE <input type="checkbox"/> TABLE																																	
(c) OUTPUT-2																																																		

Figure 13. RAM Allocation

input processing provides the carrier generation, mixing, aliasing filter and lowpass filter functions. Fig. 13(a) shows the RAM allocation during input processing. The input segment includes an efficient iteration loop, using the PAL loop counter, to process the 50-kHz functions. The carrier frequency shift and lowpass filter functions are processed at the $f_s/32$ reduced sample rate for maximum throughput efficiency.

The values of $\sin \Delta$, $\cos \Delta$, and the Bessel filter coefficients depend on the analyzer bandwidth strap selection. These values are stored in a "table" area in Fig. 13, and can be easily changed. The fixed aliasing filter coefficients are stored as constants in the microprogram itself.

Once the input processing is complete, coefficients located in the table area can be changed. This area is re-used by the output program segment to hold the scale factors for the linear-to-log conversion routine. The detector functions are processed, and the logarithmic conversion is started with the RAM allocation of Fig. 13(b). The table area is then reloaded with the interpolation coefficients (Fig. 13(c)) to complete the logarithmic conversion. Shaded areas in Fig. 13 provide temporary data and flag storage for the routines.

The microprogram samples the strap settings and loads the table area with the appropriate coefficients for input processing. The detector filter coefficient (b_1) is also determined and loaded. The input processing is then repeated. This sequence repeats indefinitely. The coefficient loading technique makes efficient use of RAM capacity while eliminating elaborate jump sequences. All coefficient table updates are processed at the minimum sample rate for best efficiency.

The PAL controllers simplify the microprogram. A PAL provides hardware iteration loops. The AU controller eliminates wasteful "NO-OP" instructions otherwise required to allow completion of AU operations. The input control PAL simplifies handshaking with the input logic. With the benefit of the PAL controllers, the analyzer microprogram easily fits into the 256-instruction capacity of the PROMs.

Performance

Table IV summarizes the performance of the spectrum analyzer. The performance figures indicate that the 67516-based analyzer is quite useful for high-fidelity audio work. Analyzer performance using other Monolithic Memories multipliers is given in the table. Performance of a spectrum analyzer using the Intel 2920 (Ref. 6) is also shown as an example of the capability of a single-chip processor.

The bipolar approach provides substantial performance improvement, at the expense of increased package count. The 67516-based analyzer provides over six times the input bandwidth of the 2920-based analyzer, even though the 2920 multiply time is only 20 percent slower than the 67516. The performance difference is due to restrictions in the 2920 architecture that preclude generation of precision sinusoids at high speed and limit the ability to use sample rate reduction. These restrictions require use of a mixing scheme that must operate with bandwidths much less than the Nyquist requirements, and also precludes sample rate reduction. The performance advantages of a flexible architecture are clearly shown in this case.

Other single-chip DSPs present different problems. Unpublished analyzer benchmarks using the AMI S2811 reveal that while sampling rates in excess of 40 kHz are possible with this part, the internal 12-bit word length of the S2811 limits dynamic range to about 46 dB due to limit cycle noise. The 2920's 25-bit word length avoids this difficulty, but the 9-bit 2920 A/D resolution again limits dynamic range to 48 dB. To achieve both the high speed and performance required for high-fidelity analysis, therefore, the flexibility of the bipolar approach is required.

The bipolar flexibility is apparent when considering options available with the analyzer architecture. Use of faster bipolar multipliers can provide input bandwidths over 125 kHz (see Table IV), at the expense of a higher package count. Pipeline techniques can be added to achieve even greater bandwidth. Dynamic range can be improved by using a wider data word, again at the cost of additional packages. Designers can trade off performance versus complexity to get the job done.

Analyzer sweep rate and accuracy can also be traded off. The sweep rate for the 67516 analyzer is $(BW/n) \times 1.56$ kHz/sec, where BW is the bandwidth and n is given by Eq. (16). For example, the sweep rate with a 500-Hz bandwidth can be increased to 97.6 kHz/sec by relaxing the accuracy requirement to 1 dB. The resulting 4.88-Hz refresh rate for a full 20-kHz sweep is readily viewed on an oscilloscope.

The bipolar analyzer combines the high performance of MSI components with the flexibility and programmability of single-chip processors. The ability to customize the architecture provides yet another degree of flexibility. PAL controllers can be used to implement the analyzer in less board space than would be required with bit-slice components. The Monolithic Memories' components provide an efficient, cost-effective alternative in many digital signal processing applications.

TABLE IV. ANALYZER PERFORMANCE SUMMARY

ANALYZER IMPLEMENTATION				
	MMI 67508	MMI 67516	MMI 67558	INTEL 2920
f_s (maximum)	21 kHz	56 kHz	356 kHz	16 kHz
Input bandwidth	7.5 kHz	20 kHz	125 kHz	3 kHz
Resolution	20-500 Hz	20-500 Hz	20-500 Hz	100 Hz
Accuracy	0.1 dB	0.1 dB	0.1 dB	1 dB
Dynamic range	60 dB	60 dB	60 dB	48 dB
Sweep rate (kHz/sec)	.4 – 9.8	.4 – 9.8	.4 – 9.8	6
Packages	34	34	50	2

Monolithic Memories

Americas

Monolithic Memories

1165 East Arques Avenue
Sunnyvale, CA 94086
Phone (408) 739-3535
Telex (910) 339-9229

France

Monolithic Memories France S.A.R.L.

Silic 463
F 94613 Rungis Cedex
France
Phone 1-6874500
Telex 202146
Fax 1-6876825

Japan

Monolithic Memories Japan KK

4-5-15, Sendagaya
Shibuya-Ku
Tokyo 151
Japan
Phone 3-4039061
Telex 781-26364
Fax 3-4040570

England

Monolithic Memories Ltd.

Lynwood House
1 Camp Road
Farnborough, Hampshire
United Kingdom GU146EN
Phone (0252) 517431
Telex 858051 MONO UKG
Fax (0252) 43724

Germany

Monolithic Memories, GmbH

Mauerkircherstr 4
D 8000 Munich 80
West Germany
Phone 89-984961
Telex 524385
Fax 89-983162

Monolithic Memories reserves the right to make changes in order to improve circuitry and supply the best product possible.

Monolithic Memories cannot assume responsibility for the use of any circuitry described other than circuitry entirely embodied in their product. No other circuit patent licenses are implied.

Printed in U.S.A./TL-109/3-81